



Certifying the interoperability of RDF database systems

Karima Raves, Julien Nauroy, Cécile Germain

► To cite this version:

Karima Raves, Julien Nauroy, Cécile Germain. Certifying the interoperability of RDF database systems. LDQ 2015 - 2nd Workshop on Linked Data Quality, May 2015, Portorož, Slovenia. hal-01147765

HAL Id: hal-01147765

<https://inria.hal.science/hal-01147765>

Submitted on 1 May 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - NoDerivatives| 4.0 International License

Certifying the interoperability of RDF database systems

Karima Rafes¹, Julien Nauroy², and Cécile Germain³

¹ INRIA-Saclay / BorderCloud karima.rafes@inria.fr

² INRIA-Saclay julien.nauroy@inria.fr

³ University Paris Sud and CNRS cecile.germain@lri.fr

Abstract. In March 2013, the W3C recommended SPARQL 1.1 to retrieve and manipulate decentralized RDF data. Real-world usage requires advanced features of SPARQL 1.1. recommendations As these are not consistently implemented, we propose a test framework named TFT (Tests for Triple stores) to test the interoperability of the SPARQL endpoint of RDF database systems. This framework can execute the W3C’s SPARQL 1.1 test suite and also its own tests of interoperability. To help the developers and end-users of RDF databases, we perform daily tests on Jena-Fuseki, Marmotta-KiWistore, 4Store and three other commercial databases. With these tests, we have built a scoring system named SPARQLScore and share our results on the website <http://sparqlscore.com>.

Keywords: Linked Data Quality, interoperability, SPARQL

1 Introduction

The current W3C recommendation SPARQL 1.1 has been published in its final form in May 2013 [11]. The W3C has defined tests for the compliance of RDF databases to this recommendation. Most editors of RDF databases claim to support this latest recommendation but the official implementation report for SPARQL 1.1 [10] shows that none of them pass all the official W3C tests. Moreover, software vendors explicitly forbid the disclosure of test compliance results. There exists some reference *performance* benchmarks, e.g., the Berlin Sparql Benchmark [2], and for *ontology support*, e.g., the University Ontology Benchmark [5] (even though it may be argued that this second type of benchmarks is not representative of real-world applications [13]). Surprisingly enough, it seems that there exists no exhaustive and up-to-date benchmarking facility of the W3C tests for evaluating RDF databases with respect to *interoperability*.

Thus, predicting beforehand the support of a particular SPARQL 1.1 feature in a given RDF database was impossible. This is generally damaging the deployment of the Semantic Web, and has particularly pernicious consequences in scientific research ecosystems.

In the CDS (Center for Data Science) project of Paris-Saclay University, we develop an integrated framework that offers a seamless facility to run and exploit exhaustive testing of RDF databases, in order to help our scientific communities

to choose the best solution to share their data. Our panel is broad: large and well-organized communities such as High Energy Physics (CERN experiments) as well as local communities that just discover the need to share beyond short-lived experiments, and many more; it includes both hard and soft science.

Our current TFT workflow automatically compiles, deploys and tests every night several hand-picked RDF databases from their sources as well as one SPARQL endpoint offered by a software vendor. It maintains a database of test results accessible from a web interface. The workflow will shortly be integrated within a platform as a service (PaaS), where TFT will be used to evaluate the conformity of a virtual image hosting an open source RDF database to the latest SPARQL standards, thus providing the scientific end users with critical information for a better informed choice of database based on their needs (performance, support for a particular ontology, etc.), including SPARQL federated queries. The vendors will also be able to propose virtual machines including their own RDF database system, which will then be automatically evaluated using the TFT software before being proposed to researchers.

2 Innovation

2.1 Is interoperability impossible?

The Semantic Web, or Web of Data, aims among other things to share readable information between humans and machines. When this exchange will be possible, new machines will be born to help the humans to use all the information on the Web. This huge amount of information is already unusable by humans but the majority of the machines are also unable of handling it alone.

The machines on the Web become specialized: collector, calculator, semantic parser, databases, etc. The availability of APIs with the Webservice technology was the first response to the need to communicate between machines. Unfortunately, there are as many APIs as developers. This heterogeneity makes impossible the implementation of autonomous agents able to discover and consume the Web data, as it had set a simple API and a unique protocol. Enabling such agents is the aim of SPARQL, by making them capable to discover the data across the Web without downloading them beforehand. This is also a major issue for the Web of Things, where every object becomes a potential Web Agent.

Lack of interoperability causes two complications that are critical to widespread adoption of RDF by large and organized scientific communities such as HEP (High Energy Physics): migration between databases and their upgrades and the development of agents, as experienced manpower is scarce.

Total interoperability might still be a long way to go. Should we wait until it happens? Instead, a medium term strategy can take into account the fact that, in practice, the end-users could cope with some limitations. However, their tradeoffs are different, thus a first-order requirement is to be able to precisely assess the strengths and flaws of databases with respect to interoperability. The tool for evaluating interoperability did not exist. **TFT answers this critical need.**

2.2 The last version is always the better

Interoperability is not enough for scientific communities. The most advanced ones want to use the latest database technology (inferences, velocity, clustering and so on). These innovations are rarely available in the stable versions of databases before several months or even several years. The unstable versions are often available for free download and researchers can install these latest versions very quickly with tools like Git. Moreover, for the small communities, the compromise between compliance to standards and cutting-edge performance is often arbitrated more or less blindly in favor of the latter. The TFT software provides a simple solution to make a better informed decision, creating an incentive for selecting the more interoperable technology within the user requirements.

2.3 IaaS for the researchers

For a Chief Information Officer (CIO) in the academic world providing services to multiple small and poorly organized scientific communities, the condition of interoperability is not enough to deploy a software in an information system. The CIOs have strong Quality of Service constraints (QoS). The solution of CIO is to offer an IaaS (Infrastructure as a Service, typically a local cloud). With this IaaS, the researcher can create or disappear a virtual machine in a few clicks, and can install her preferred tools without bothering with QoS, security and interoperability. After evaluation of the research results by the peers, the resources may disappear fairly quickly because the corresponding data are still rarely integrated in a long term archiving plan.

These careless methods are doomed. The scientific agencies are enforcing the requirement to linking data to results, with reproducibility as the ultimate goal. Nobody can replace the researchers to save their work and share their findings, with mechanisms such as the Digital Object Identifier (DOI) System [7]. However, our PaaS will help: it will facilitate the transition from small ephemeral silos to permanent repositories within clouds, without sacrificing the agile development that is essential to a significant part of real-world good research.

2.4 Wrap-up

The TFT software certifies the last version of RDF database system using a continuous delivery workflow. By providing seamless choice of the last best interoperable databases, our innovation facilitates data sharing within and across scientific communities. Beyond selection, our PaaS contributes to the advent of reproducible science.

3 Detailed features and function

3.1 Overview

The TFT has 4 parts: 1) upload the benchmarks into our RDF database, 2) run the tests, 3) compute a score for each database systems, and 4) share the results in RDF via SPARQL.

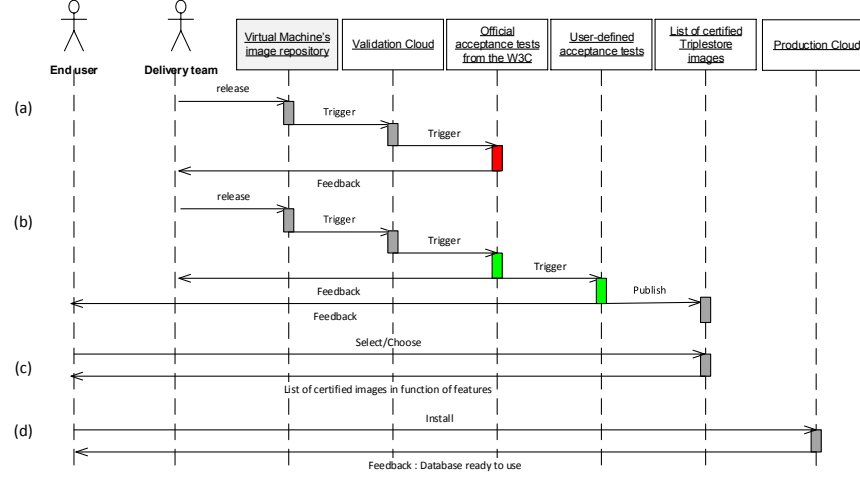


Fig. 1. Certification workflow

Currently TFT offers a score on interoperability of software and also provide a RDF database of detailed test results. In the near future, these results can power tools in the cloud that will facilitate the provision of solution of latest generation databases for the researchers and will be maintained by CIOs, by ensuring interoperability of data, and will facilitate their work of preserving data by being able to simply migrate data from one system to another. TFT can also be integrated into continuous integration environments of database editors, in order to improve their products and the CIOs can check the RDF database system in their environments. Fig. 1 summarizes the workflow. After a new release of a given RDF database, software is made available in the form of a virtual appliance, the image is run on a validation cloud and a set of tests is performed over this new database instance. According to the results of the tests, the validation can either (a) fail and a feedback is provided to the appliances publisher or (b) user-defined tests are run to validate their specific needs.

3.2 The benchmarks

Upload the tests. For now, there are two collections of tests: the SPARQL 1.1 test suite (453 tests) [9], and a test suite (6 tests) from the GO [3] project. The file `config.ini` defines the collection of tests and can be extended when necessary. Each collection of tests has a separate folder in the project *TFT-tests* on GitHub [12]. Each folder contains a file named `manifest-all.ttl` containing pointers to the files related to the test, according to the W3C format. Fig. 2 shows an example of a test with a federated query. This test needs to have two remote endpoints to execute the query. The file `pbs.ttl` contains the input to be loaded into the first remote endpoint and the file `bdi.ttl` contains the input to be loaded into

```

: test_10    rdf:type mf:QueryEvaluationTest ; #Type of test
mf:name      "Query to calculate ERT-ART" ;
dawgt:approval dawgt:Approved;
#Type of tool to run the test
mf:feature sd:BasicFederatedQuery ;
mf:action
    [ qt:query <q10.rq> ;

      qt:serviceData [
        qt:endpoint <http://example1.org/sparql> ;
        qt:data      <pbs.ttl>
      ];
      qt:serviceData [
        qt:endpoint <http://example2.org/sparql> ;
        qt:data      <bdii.ttl>
      ]
    ] ;
mf:result    <q10.srx> .

```

Fig. 2. A test for a federated query in the project TFT-tests [12].

```

[SERVICE]
endpoint["http://example.org/sparql"] = "http://o1.in2p3.fr/sparql/"
endpoint["http://example1.org/sparql"] = "http://o2.in2p3.fr/sparql/"
endpoint["http://example2.org/sparql"] = "http://o3.in2p3.fr/sparql/"

```

Fig. 3. During the tests the remote endpoints are replaced by their real URLs (file config.ini in the software TFT [8])

```

git clone --recursive https://github.com/BorderCloud/TFT.git
cd TFT

#upload the test suites in a database
./tft-testsuite -a -t fuseki -q http://example.com:3030/tests/query \
               -u http://example.com:3030/tests/update

#execute the test suites on the other database and save the results.
./tft \
  -t fuseki -q http://example.com:3030/tests/query \
            -u http://example.com:3030/tests/update \
  -tt fuseki -tq http://127.0.0.1/ds/query \
            -tu http://127.0.0.1/ds/update \
  -o ./junit \
  -r ${BUILD_URL} \
  --softwareName=Fuseki \
  --softwareDescribeTag=v${VERSIONFUSEKI} \
  --softwareDescribe="${BUILD_TAG}#${FILEFUSEKI}"

```

Fig. 4. This script downloads TFT, uploads, passes and saves the tests and the results in a RDF database.

the second remote endpoint. During the tests, TFT will replace the URIs of the remote endpoints with the URI contained in the file `config.ini` (Fig. 3).

Pass the tests. We use Jenkins, a continuous integration server, because the database systems are often Open Source and in a Git repository. Fig. 4 shows an

example of a script execution by our continuous integration server. It follows the same workflow for each test: 1) delete all data from the main test database and the remote test database(s); 2) load initial data to define the initial state in the main database and the remote database(s); 3) run the tests in the main database; in case of federated queries, the main database is responsible for contacting the remote ones (this is the normal behavior of federated queries) ; 4) Monitor the response to the test and/or control the final state obtained in the databases. After the tests have been run, the script `tft` saves and shares the results.

Compute a score. Choosing a particular weight for each of our 459 tests would be highly debatable. We calculate a simple global score for each RDF database system: one point is given for each passed test. The script `tft-score` calculates this score and shares these scores and the results of tests.

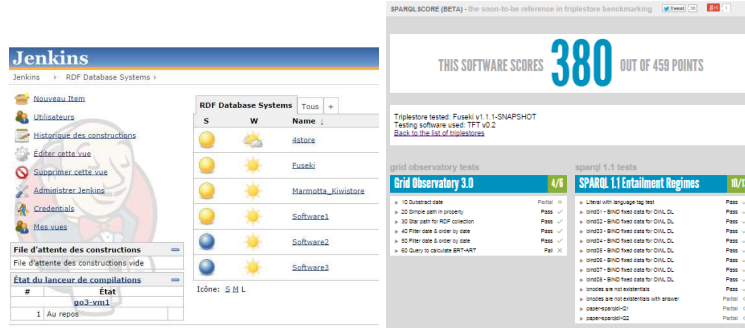


Fig. 5. Jenkins and SparqlScore use the test results

3.3 Share the results

After the compliance tests have been run, we share three results with various actors

With the editors. TFT creates a report in JUnit format compliant with the Jenkins software (Fig. 5, left). Jenkins can check the last push in the Git repository about a software and can give a feedback to developers in real-time. If a software editor integrates TFT in its Jenkins server, he will also be able to reject automatically the last delivery if a test shows a regression of interoperability. An example of tests is available in the project TFT-tests on GitHub [12] and the *developers* can see the tests of the end-users and can reproduce the same tests. They can also add their own tests easily.

With the machine. After the compliance tests have been run, TFT generates two reports: one in JUnit format for our own Jenkins server and one in RDF/EARL (Evaluation and Report Language) format [1]. The report in EARL format is saved in an RDF database exposing a SPARQL endpoint. Thus another *machine* can check easily the compliance of RDF database systems. So, we can integrate new software almost in real-time following the last deliveries of

developers in our PaaS and check the compatibility. The continuous integration platform can alert if there is a regression in the software and the machine can detect the improvements, propose the last best stable databases and migrate automatically the database in the best last stable solutions for the researchers.

With the end-users The **SparqlScore.com** website, Fig. 5 (right), illustrates the reuse of the test results and database scores. In order to improve the user's experience of the website and to relieve our database, we use the Smarty [6] library to cache the results of the SPARQL queries to build the report in HTML5. With this website, the *end-users* can see the real interoperability of database and in the future, others indicators.

4 Design choices

We integrate the linked data technologies where the input are the tests in the turtle format with the ontology defined by the SPARQL 1.1 WG; the output is a RDF database that is fed by a SPARQL Update query after each test.

This design offers the possibility to write quickly a new test and everybody can propose a new test or fork the tests via a project in the GitHub's Service [12].

The TFT software is under a Creative Commons Attribution-ShareAlike 4.0 International License. The aim of this license is to share the same software to test and compare objectively the databases on the market. TFT and TFT-tests (the collections of tests) are available via their repositories [8][12].

The SparqlScore software is also available via its repository [12] and everybody can read the last results with our continuous integration platform on the website <http://sparqlscore.com/> (Fig. 5).

5 Conclusion

Benchmarking without testing the protocol is insufficient

The SPARQL update protocol is not identical across databases. Developing a Web agent with SPARQL without knowing the exact server software is quite difficult. The reason is simple: the protocol concerning update queries is fuzzy in the SPARQL recommendation thus each database implements a different flavor of the protocol. TFT can test five RDF databases because we have to implement the specificities of each database in order to execute the same queries.

An open benchmark is possible and can help to converge

Very quickly after the launch of the sparqlscore.com website, four vendors contacted us to include their software in our tests and three accepted to open their results. Three vendors have a specifically set-up a SPARQL endpoint for our tests. The editors started to discuss how to interpret the recommendation and several fixed some interoperability problems.

The SPARQL 1.1 is a recommendation but not the tests

The official test suite is a great job as a starting point. Each difference in the result of a SPARQL query in an RDF database is an obstacle to the deployment of Linked Data in public institutions or a simple company. But interoperability is

not an option in the Linked Data, it's the first aim. Moreover, a lot people want use the Linked Data's technologies and the classical access control problems are resolved separately by the different editors. The editors have to diverge from the recommendation in order to resolve the security needs of their customers.

Help to create a really interoperable ecosystem ?

The W3C has launched the Test the Web Forward[4] initiative in 2013. The main goal of this action is to incentive web developers to ensure better interoperability on the Web. For the moment, this initiative tests the technologies related to Web browsers. With the TFT solution, the developers can also propose new tests as like "Test the Web Forward". So, it would make sense to extend the "Test the Web Forward" to SPARQL to begin and after to continue with the other technologies of Semantic Web.

We hope TFT can be useful for the possible evolution of the "Test the Web Forward" initiative to create a really interoperable ecosystem for the Linked Data.

Acknowledgments

This work has been partially funded by the TIMCO project, by the Paris-Saclay Center for Data Science (funded by the IDEX Paris-Saclay, ANR-11-IDEX-0003-02), and by France Grilles.

References

1. Abou-Zahra, S., W3C/WAI: Evaluation and Report Language (EARL) 1.0 Schema (2011)
2. Bizer, C., Schultz, A.: The Berlin SPARQL benchmark. *Int. J. On Semantic Web and Information Systems* 4(2), 1–24 (2009)
3. Germain-Renaud, C., al.: The grid observatory. In: *Cluster, Cloud and Grid Computing (CCGrid)*, 11th IEEE/ACM Int. Symp. on. pp. 114–123. IEEE (2011)
4. Langel, T.: *Testing the open Web platform* (2013)
5. Ma, L., Yang, Y., Qiu, Z., Xie, G., Pan, Y., Liu, S.: *Towards a complete OWL ontology benchmark*. Springer (2006)
6. New Digital Group, Inc: *What is Smarty?* (2014)
7. Paskin, N.: Digital object identifier (doi) system. *Encyclopedia of library and information sciences* 3, 1586–1592 (2008)
8. Raves, K.: *Repository Git of software TFT* (2014)
9. W3C SPARQL Working Group: *SPARQL1.1: Test case structure* (2012)
10. W3C SPARQL Working Group: *Official implementation report for SPARQL 1.1* (March 2013)
11. W3C SPARQL Working Group: *Recommendations of the W3C : SPARQL 1.1 (Protocol and RDF Query Language)* (March 2013)
12. W3C SPARQL Working Group and The grid observatory: *Repository git TFT-tests with the test suite of SPARQL1.1 and Grid Observatory* (2014)
13. Weithöner, T., Liebig, T., Luther, M., Böhm, S.: *Whats wrong with OWL benchmarks*. In: *Proc. of the Second Int. Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2006)*. pp. 101–114. Citeseer (2006)